CPSC 457

Assignment #1

Ahmad Almasri - 30114233

Question 1

t3.txt

```
ahmad.almasri3@gfx03-7:~/cpsc457/a1$ time python3 palindrome.py < t3.txt
Longest palindrome: ___o.O.o___

real    0m0.025s
user    0m0.015s
sys     0m0.005s
```

t4.txt

```
ahmad.almasri3@gfx03-7:~/cpsc457/a1$ time python3 palindrome.py < t4.txt
Longest palindrome: redder

real    0m0.260s
user    0m0.241s
sys     0m0.011s
```

t3.txt

```
ahmad.almasri3@gfx03-7:~/cpsc457/a1$ time ./slow-pali < t3.txt
Longest palindrome: ___o.O.o___

real    0m0.009s
user    0m0.002s
sys     0m0.003s
```

t4.txt

```
ahmad.almasri3@gfx03-7:~/cpsc457/a1$ time ./slow-pali < t4.txt
Longest palindrome: redder

real    0m2.454s
user    0m1.116s
sys     0m1.334s
```

t3.txt		
	C++	**Pyhon**
User	0.002s	0.015s
Kernel	0.003s	0.005s

t4.txt		
	C++	**Pyhon**
User	1.116s	0.241s
Kernel	1.334s	0.011s

In the first place, there is a main difference between C++ and Python. C++ is considered to be a **compiled language**. However, Python is a **compiled and interpreted** one.

Compiled language means that we compile the source code to a **machine language** (binary file). The binary file is called the **executable file**. Because of the binary file, running the c++ code would be very **fast** because the code would be compiled ahead of time. **However,** having a compiled file requires us to have different versions for each OS.

Python is an **intermediate language** (Bytecode). Where, the source code of python being compiled to a **Bytecode**. The Bytecode would be interpreted line by line, while executing the program. For example, the interpreter for Python is **PVM** and JAVA is the **JVM**.

In case of t3.txt, C++ was faster than Python. I think, the reason of that result is that the size of the text file is small. On the other hand, **the t4.txt** was bigger in size. Python reads the whole content by using less number of system calls.

In general, C++ is faster than Python, but the way that we write the code must be efficient to take advantage of the language either C++ or Python.

Question 2

```
ahmad.almasri3@gfx-ta3:~/cpsc457/a1$ ./dup.py 2000000000 < t4.txt | time ./fast-pali
Longest palindrome: redder
24.34user 0.27system 0:25.53elapsed 96%CPU (0avgtext+0avgdata 3980maxresident)k
0inputs+0outputs (0major+383minor)pagefaults 0swaps
ahmad.almasri3@gfx-ta3:~/cpsc457/a1$
```

Question 3

```
ahmad.almasri3@gfx-ta3:~/cpsc457/a1$ strace -c ./fast-pali < t4.txt
Longest palindrome: redder
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 50.37    0.001309          72        18           read
 21.43    0.000557          11        48        43 openat
 12.31    0.000320          14        22           mmap
  3.66    0.000095          13         7           mprotect
  3.27    0.000085          10         8         7 stat
  2.27    0.000059           8         7           lseek
  1.77    0.000046           7         6           fstat
  1.69    0.000044           8         5           close
  1.08    0.000028           9         3           brk
  0.85    0.000022          22         1           munmap
  0.65    0.000017          17         1         1 access
  0.65    0.000017           8         2         1 arch_prctl
  0.00    0.000000           0         1           write
  0.00    0.000000           0         1           execve
------ ----------- ----------- --------- --------- ----------------
100.00    0.002599          19       130        52 total
```

```
ahmad.almasri3@gfx-ta3:~/cpsc457/a1$ strace -c ./fast-pali < t3.txt
Longest palindrome: ___o.O.o___
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 28.41    0.000125           2        48        43 openat
 24.32    0.000107         107         1           execve
 17.05    0.000075           3        22           mmap
  6.59    0.000029           2        13           read
  5.68    0.000025           3         7           mprotect
  4.09    0.000018           2         8         7 stat
  2.95    0.000013           1         7           lseek
  2.73    0.000012           2         6           fstat
  2.27    0.000010           2         5           close
  1.59    0.000007           7         1           munmap
  1.59    0.000007           2         3           brk
  0.91    0.000004           4         1           write
  0.91    0.000004           4         1         1 access
  0.91    0.000004           2         2         1 arch_prctl
------ ----------- ----------- --------- --------- ----------------
100.00    0.000440           3       125        52 total
```

```
ahmad.almasri3@gfx-ta3:~/cpsc457/a1$ strace -c ./slow-pali < t4.txt
Longest palindrome: redder
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
100.00    9.592183           1   5767205           read
  0.00    0.000075           1        48        43 openat
  0.00    0.000053           7         7           mprotect
  0.00    0.000052           2        22           mmap
  0.00    0.000009           1         7           lseek
  0.00    0.000008           8         1           write
  0.00    0.000008           1         6           fstat
  0.00    0.000006           1         5           close
  0.00    0.000004           4         1           munmap
  0.00    0.000002           0         3           brk
  0.00    0.000002           1         2         1 arch_prctl
  0.00    0.000000           0         8         7 stat
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           execve
------ ----------- ----------- --------- --------- ----------------
100.00    9.592402           1   5767317        52 total
```

```
ahmad.almasri3@gfx-ta3:~/cpsc457/a1$ strace -c ./slow-pali < t3.txt
Longest palindrome: ___o.O.o___
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 32.74    0.000204         204         1           execve
 20.55    0.000128           2        48        43 openat
 16.69    0.000104           2        50           read
 12.20    0.000076           3        22           mmap
  4.01    0.000025           3         7           mprotect
  3.21    0.000020           2         8         7 stat
  2.09    0.000013           2         6           fstat
  2.09    0.000013           1         7           lseek
  1.61    0.000010           2         5           close
  1.28    0.000008           2         3           brk
  1.12    0.000007           7         1           munmap
  0.96    0.000006           6         1           write
  0.80    0.000005           5         1         1 access
  0.64    0.000004           2         2         1 arch_prctl
------ ----------- ----------- --------- --------- ----------------
100.00    0.000623           3       162        52 total
```

fast-pali.cpp is way faster than slow-pali.cpp. The main reason is the reduced number of SYS calls that we are performing to read the data. The size of the buffer in the slow version is only 1 Byte; however, the fast version is 1 MB.

Part b

```
ahmad.almasri3@gfx-ta3:~/cpsc457/a1$ strace -c python3 palindrome.py < t4.txt
Longest palindrome: redder
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 17.61    0.000200           1       175        47 stat
 13.91    0.000158           1       141        76 openat
 12.24    0.000139           0       788           read
  9.77    0.000111           1       100           fstat
  8.98    0.000102           1        58           mmap
  8.80    0.000100           9        11           mprotect
  7.66    0.000087           1        68           close
  7.22    0.000082           5        16           getdents64
  3.26    0.000037           0        42         2 lseek
  2.46    0.000028           1        18        11 ioctl
  2.38    0.000027           0        58           brk
  1.14    0.000013          13         1           lstat
  0.88    0.000010           3         3           dup
  0.79    0.000009           4         2           munmap
  0.53    0.000006           2         3         2 readlink
  0.53    0.000006           3         2           futex
  0.35    0.000004           0        68           rt_sigaction
  0.26    0.000003           1         3           fcntl
  0.26    0.000003           3         1           getrandom
  0.18    0.000002           2         1           rt_sigprocmask
  0.18    0.000002           1         2         1 arch_prctl
  0.18    0.000002           2         1           set_tid_address
  0.18    0.000002           2         1           set_robust_list
  0.18    0.000002           2         1           prlimit64
  0.09    0.000001           1         1           getcwd
  0.00    0.000000           0         1           write
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           getpid
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         1           sysinfo
  0.00    0.000000           0         1           getuid
  0.00    0.000000           0         1           getgid
  0.00    0.000000           0         1           geteuid
  0.00    0.000000           0         1           getegid
  0.00    0.000000           0         3           sigaltstack
------ ----------- ----------- --------- --------- ----------------
100.00    0.001136           0      1577       140 total
```

Part b

```
ahmad.almasri3@gfx-ta3:~/cpsc457/a1$ strace -c python3 palindrome.py < t3.txt
Longest palindrome: ___o.O.o___
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 19.79    0.000285           1       175        47 stat
 19.65    0.000283           2       141        76 openat
 14.93    0.000215          13        16           getdents64
 10.21    0.000147           2        58           mmap
 10.07    0.000145           1        84           read
  7.08    0.000102           1       100           fstat
  6.46    0.000093           1        68           close
  2.85    0.000041           0        42         2 lseek
  2.01    0.000029           2        11           mprotect
  1.74    0.000025          25         1           lstat
  1.18    0.000017           0        18        11 ioctl
  0.83    0.000012           0        18           brk
  0.62    0.000009           3         3         2 readlink
  0.35    0.000005           5         1           write
  0.28    0.000004           0        68           rt_sigaction
  0.28    0.000004           1         3           fcntl
  0.28    0.000004           1         3           sigaltstack
  0.28    0.000004           2         2         1 arch_prctl
  0.21    0.000003           3         1         1 access
  0.21    0.000003           1         3           dup
  0.21    0.000003           3         1           getcwd
  0.21    0.000003           1         2           futex
  0.14    0.000002           2         1           getrandom
  0.07    0.000001           1         1           getuid
  0.07    0.000001           1         1           getegid
  0.00    0.000000           0         2           munmap
  0.00    0.000000           0         1           rt_sigprocmask
  0.00    0.000000           0         1           getpid
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         1           sysinfo
  0.00    0.000000           0         1           getgid
  0.00    0.000000           0         1           geteuid
  0.00    0.000000           0         1           set_tid_address
  0.00    0.000000           0         1           set_robust_list
  0.00    0.000000           0         1           prlimit64
------ ----------- ----------- --------- --------- ----------------
100.00    0.001440           1       833       140 total
```

Comparing it with the python code, the c++ version is faster. First, c++ is faster than python in general. Second, the size of the buffer significantly reduced the SYS calls for fast-pali.cpp.